LA-UR-19-29909
Approved for public release; distribution is unlimited.

Title: Eulerian Applications Project - xRage The xRage Equation of State: Introduction and Basic Usage

Author(s): Grove, John W.

Intended for: Report

Issued: 2019-09-30

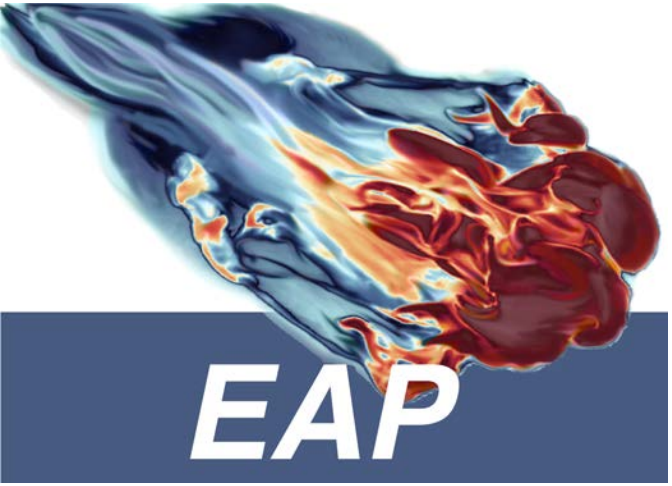
Los Alamos
NATIONAL LABORATORY
EST.1943

Delivering science and technology
to protect our nation
and promote world stability

NNSA
National Nuclear Security Administration

Managed by Triad National Security, LLC for the U.S. Department of Energy's NNSA

Eulerian Applications Project - xRage



The xRage Equation of State: Introduction and Basic Usage

John W. Grove
CCS-2

September 25, 2019

Los Alamos NATIONAL LABORATORY EST.1943

NNSA National Nuclear Security Administration

Managed by Triad National Security, LLC for the U.S. Department of Energy's NNSA

Thermodynamics 101

Basic Equilibrium Thermodynamics

- The Euler system is not closed as the conservation principles create fewer equations than unknowns
- The system is closed by providing a relation between the thermodynamic variables
  - mass fractions, density, pressure, specific internal energy, temperature, specific entropy
- Most commonly for single component flows this closure is provided by giving the pressure as a function of the specific internal energy and specific volume (or density): $P = P(V, e)$
  - This relation is called an incomplete equation of state
  - For a perfect gas we have $P = (\gamma - 1)\rho e$
- The general case is considerably more complex
  - Non perfect gas behavior – solids, liquids, ionizing gases for example
  - Equilibrium conditions are one choice
    - Pressure/Temperature
    - Volume/Temperature
  - Other closures usually required additional dynamic equations
    - Thermal Isolation – component specific entropy advection
    - Uniform Compression – component volume fraction advection
- In this set of notes we will focus on single EOS models and equilibrium mixtures

Incomplete Equations of State

- For single component flows an incomplete equation of state is usually sufficient for simple hydrodynamics
- The hydro update simply recomputes the pressure as a function of the current specific volume and specific internal energy as needed
- The isentropic bulk modulus and hence the sound speed as well as the Grüneisen exponent can be directly computed from the incomplete equation of state

$$\Gamma = V \left.\frac{\partial P}{\partial e}\right|_V , \rho c^2 = -V \left.\frac{\partial P}{\partial V}\right|_e + \Gamma P$$

- The Euler system is hyperbolic when the isentropic bulk modulus is non-negative (strict hyperbolicity requires it to be positive).
- Temperature and specific entropy are not defined for an incomplete equation of state
- Constant specific entropy curves are obtained as solutions to the ordinary different equations

$$\frac{de}{dV} = -P(V, e), \qquad e(V_0) = e_0$$

Complete Equation of State

- A complete equation of state specifies the temperature and by extension the specific entropy as a function of the thermodynamics
  - A common method is to give pressure and specific internal energy as functions of specific volume (or density) and temperature

$$P = P(V,T), e = e(V,t)$$

  - A SESAME 301 table is an example (they use density instead of specific volume)
- A thermodynamically consistent equation of state corresponds to the existence of a thermodynamic free energy
  - Helmholtz Free Energy: $F(V,T), dF = -SdT - PdV, P = -\left.\frac{\partial F}{\partial V}\right|_T \; S = -\left.\frac{\partial F}{\partial S}\right|_V$
  - Specific Internal Energy: $e(V,S), de = TdS - PdV, P = -\left.\frac{\partial e}{\partial V}\right|_S \; T = \left.\frac{\partial e}{\partial S}\right|_V$
  - Specific Enthalpy: $H(P,S), dH = TdS + VdP, V = \left.\frac{\partial H}{\partial P}\right|_S \; T = \left.\frac{\partial H}{\partial S}\right|_V$
  - Gibb's Free Energy: $G(P,T), \;\; dG = -SdT + VdP, V = \left.\frac{\partial G}{\partial P}\right|_T \; S = -\left.\frac{\partial G}{\partial T}\right|_P$

Relation Between Free Energies

- All of the free energies are related to one another and can be transformed via suitable Legendre Transforms
  - Actually the invertibility of the transforms depends on the thermodynamic stability of the free energies
  - $e(V,S) = \sup_T [F(V,T) + TS]$, $F(V,T) = \inf_S [e(V,S) - TS]$
  - $G(P,T) = \inf_{V,S} [e(V,S) - TS + PV]$, $e(V,S) = \sup_{P,T} [G(P,T) + TS - PV]$
  - $H(P,S) = \inf_V [e(V,S) + PV]$, $e(V,S) = \sup_P [H(P,S) - PV]$
- Invertibility of these transforms depends on the thermodynamic stability of the various free energies, which is a statement about their convexity or concavity
- $e(V,S)$ is jointly convex, $G(P,T)$ is jointly concave
- $F(V,T)$ is convex in V concave in T
- $H(P,S)$ is concave in P convex in S
- Thermodynamic stability is sufficient for the hyperbolicity of the Euler system

Amagat Mixtures

- The Amagat mixture law assumes that the components in a mixture are immiscible and occupy separate volumes at the microscale
  - Interfaces exist between the materials but are unresolved at the computational length scale
- The components in the mixture are in pressure and thermal equilibrium.
  - Pressure or mechanical equilibrium assumes that the wave crossing times comparable to the mixture size so that interactions between the components are sufficient to bring all of the materials to the same pressure in times smaller than the computational time step.
  - Thermal equilibrium depends on the same processes and in addition assume that thermal diffusion occurs at a fast enough rate to bring everything to the same temperature within a time step.
    - This assumption is not valid for high speed and shock dominated flows, but has proven to be useful in practice
    - Basically thermal discontinuities are resolved on the computational mesh.
- A variety of models have been used to relax these assumptions
  - Generally they require additional dynamic partial differential equations to model the non-equilibrium flow.

Dalton Mixtures

- An alternative mixing model is Dalton or molecularly mixed cells
- Activated by the input file option “daltonmix=.true.”
- Assumes that all materials have the same volume and temperature.
- The pressure is the sum of the partial pressures of the components.
- For perfect gas mixtures, the Amagat and Dalton mixture laws yield the same effect cell pressure and temperature
  - Not true for general EOS models and is the source of endless confusion
- The choice of the mixture model is problem dependent and requires a value judgement on the nature of flow of interest
  - General guidance
    - gaseous mixture - Dalton
    - Solids and liquids – Amagat
    - Mixtures of liquids, solids, and gases – neither is strictly correct and non-equilibrium models may be needed
    - In practice Amagat works pretty well for many cases

Well! That's the Theory. What's the Practice?

- The generic theory outlined in the previous slides is all well and good.
- But in reality our EOS treatments are subject to may practical limitations.
- Incomplete verses Complete equations of state
  - Temperature is not defined
  - A complete EOS can be derived from an incomplete EOS by solving the partial differential equation

$$\frac{\partial T}{\partial V} - P\frac{\partial T}{\partial e} = -\frac{\Gamma}{V}T$$

  - The characteristics of this scalar partial differential equation are the isentropic curves for the incomplete EOS, so providing data for the temperature along any non-isentropic curve is sufficient to determine a temperature
  - The specific entropy and hence the free energies are then found by solving

$$T\frac{\partial S}{\partial e} = 1, T\frac{\partial S}{\partial V} = P(V, e)$$

All Well and Good. What else?

- In practice, a given equation of state, even when its mathematical formulation is complete by not be either consistent or stable
- Depending on how the temperature was defined, it is quite possible that the isentropic bulk modulus is non-negative but the isothermal bulk modulus is negative (EOS is not stable). In general the isentropic bulk modulus is greater than the isothermal bulk modulus. Indeed for a consistent EOS we have the relation

$$\rho c^2 = \rho c_T^2 + \rho \Gamma^2 C_V T$$

  - Here ρc_T^2 is the isothermal bulk modulus, $\Gamma = V \left.\frac{\partial P}{\partial e}\right|_V$ is the Grüneisen exponent, and C_V is the specific heat a constant volume (non-negative for a stable EOS)
- The model may not be consistent and hence has no associated free energy. In this case formulas like the above related the two types of bulk moduli are not even valid.
  - The use of the above formula with SESAME data often produces too large a value for the sound speed.

What's Your Point?

- The point here is that theoretical assumptions about thermodynamics may not always be satisfied for a given equation of state model.
- One must be careful about applying these assumptions in a numerical scheme.
- For instance. For an incomplete equation of state, the hyperbolic characteristics of the Euler system are the flow speed plus or minus c where $\rho c^2 = -V \left.\frac{\partial P}{\partial V}\right|_e + VP \left.\frac{\partial P}{\partial e}\right|_V$. There are a variety of equivalent expressions for the sound speed if the EOS is both consistent and stable, but these will not be the correct hyperbolic wave speed if the EOS is not consistent.
- So be careful, remember in EOS modeling no good deed goes unpunished.

The Warnings Aren't Done Yet

- Another critical issue is the except for some very special cases (basically perfect gases) the domain of an EOS model is not unrestricted.
- I think of two main domain restrictions
  - Mathematical: Outside of the domain the EOS is not consistent, not stable, or both
  - Physical: The idealized thermodynamics is okay, but the model does not represent the behavior of the material in question.
  - On would expect the physical domain to be strict subset of the mathematical domain, but even this may not be true for calibrated models
- When dealing with mixtures the situation is even worse! The valid domain is the intersection of the domains of the mixture components and it could even be that this intersection is the empty set.

How the Use the xRage Equation of State Package

- Always keeping the above warnings in mind, there are two basic ways to use the xRage EOS package
- EOS models that include mixtures
  - Primarily perfect and stiff perfect models
- General mixtures
  - Implemented by constructing a pressure-temperature inversion of the tables
- The choice of which of these two approaches are used are determined by the input parameter **keos**
  - **keos** = 0 - mixture of perfect gases
  - **keos** = 3 - pressure/temperature table lookup
  - The above two are by far the most common option
  - **keos** = -1 – Mixture of stiff perfect gases – inconsistent temperature
  - **keos** = -2 – stiff mixtures with crush curve – inconsistent temperature
  - **keos** = -3 – radiation mix, just used for debugging
  - **keos** = -4 – stiff mixtures – positive pressure – inconsistent temperature
  - **keos** = -5 – stiff mixture with ghosts – inconsistent temperature
  - **keos** = -6 – stiff mixture – thermodynamically consistent temperature
- The next series of slides will summarize each of the negative **keos** models and their initialization

Creating mixture EOS tables

- Except for the models just discussed all mixture equations of state are created by running xrage with a special input file that will generate a pressure-temperature equation of state file
- The application run will simply specify the appropriate file as the value of the input variable “eosfile=*filename*”
- There are a great many options that can be specified for the creation of the equation of state file. In many cases the defaults are sufficient but not always.
- The following is a bare bones equation of state input
  - Comments being with !

```
teos_file = "teos.out” ! Name of the EOS output file
sesfiles(1) = ”/usr/projects/data/eos/ieee64/sesame” !Location of
the sesame file containing the requested materials
numm = 2
matid(1)     = 3720        ! sesame Aluminum
eostype(2)   = 9           ! stiff gamma law
matid(2)     = 111111      ! User specific material id
matdef(16,2) = .04         ! gamma-1
matdef(30,2) = 1.e10       ! Cv
```

EOS Options

Table Creation Options: red=common, blue=recommended

Name	Type	Name	Type
teos_file	character string	pscale_neg	real
sesfiles	character string array	numprsdec_pos	integer
mat_sesfiles	character string array	numprsdec_neg	integer
numm	integer	numtev_tension	integer
matid	integer array	tevhi_tension	real
matdef	rank two real array	force_cold	logical
matident	character string array	teos_noextrapolate	logical
eostype	integer array	use_thermoderivs	logical
use_shared_memory	logical	use_eos_direct	logical
prslo	real	mxdome_itr	integer
prshi	real	do_sesame_edit	logical
numprsdec	integer	debug_eos_inv_t	logical
tevlo	real	test_tabbld	logical
tevhi	real	test_pt_inversion	logical
numtevdec	integer	list_crossed	logical
support_tension	logical	print_shared_teos_data	logical
pscale	real	debug_eos_model	logical
pscale_pos	real	high_T_emin	real

Table Creation Debugging Options

Name	Type	Name	Type
debug_eos_inv	logical	print_shared_teos_data	logical
test_tabbld	logical	debug_eos_model	logical
test_pt_inversion	logical	high_T_emin	real
list_crossed	logical		

EOS Usage Options:
red=common, blue=recommended

Name	Type	Name	Type
keos	integer	sie_min	real
eosfile	character string	enforce_sie_min	logical
eostype	integer array	rho_min	real
nummat	integer	enforce_rho_min	real
daltonmix	logical	pmin_mat	real array
matdef	rank 2 real array	pfail_mat	real array
matident	character string array		
use_eos_intrinsic_inversion	logical	void_mat	integer
VERY_COLD	real	true_void	logical
MAXIMUM_DENSITY	real	void_prs	real
MINIMUM_PRESSURE	real	void_rho	real
MAXIMUM_PRESSURE	real	void_tev	real
NONZERO_SOUND_SPEED_DENSITY_FLOOR	real	void_sie	real
ROEPS	real	VOID_CELL_TOL	real
snd_max	real		

EOS Table Evaluation Options:

Name	Type	Name	Type
MAX_ITER_TEOS	integer	use_eos_direct	logical
MAXBINARY	integer	support_tension	logical
TEOS_CONVERGENCE_TOL	real	use_thermoderivs	logical
DENSITY_TOL	real	teos_noextrapolate	logical
PRESSURE_TOL	real	ramp_num	integer
VOL_TOL	real	ramp_mat	integer array
SIE_TOL	real	ramp_alpha	real array
TEMPERATURE_TOL	real	ramp_pe	real array
MIN_P	real	ramp_pc	real array
MIN_SIE	real	ramp_pe_de	real array
use_teos_iter_5	logical	ramp_pc_de	real array
teos_secant	logical	ramp_reverse	logical array
secant_fix	logical	ramp_model	integer
do_not_push_pres	logical	ramp_coef	real array
use_eos_direct_pte	logical	ramp_power	real array
try_eos_direct_ramp	logical	ramp_pe_eng	real array
eospac_bilinear	logical	ramp_pc_eng	real array
eospac_invert_at_setup	logical	ramp_T0	real array
eospac_eos_insert_data	logical	ramp_P0	real array
use_simplified_teos	logical	ramp_rho0	real array
use_shared_memory	logical (recommend false for sesame)		

Other Options

- Non Equilibrium Options

Name	Type	Name	Type
eos_select	integer	tev_average	integer
pte_mat	integer array	single_pressure_multiple_temperature_eos	logical

- Atomic Mass/Atomic Number

Name	Type	Name	Type
abar	real	zbar	real

- Diagnostics/Debugging

Name	Type	Name	Type
debug_eos	logical	debug_teos_p_vol	logical
check_eos	logical	write_eos_domain	logical
debug_eos_direct	logical	write_eos_convergence_results	logical
check_snd_max	logical		

Obsolete Options

- Table Creation

Name	Type	Name	Type
auto_teos	logical	plot_emin	real
dorawplots	logical	plot_emax	real
plot	logical	plot_tmin	real
dolog	logical	plot_tmax	real
plot_rmin	real	plot_vmin	real
plot_rmax	real	plot_vmax	real
plot_pmin	real		
plot_pmax	real		

- Other Options

Name	Type	Name	Type
use_special_p_scaling	logical	use_old_ss	logical
MULTI_T_TOL	real	scale_p_mat	real array
FAST_MT_EOS	logical		

Tabular Implementation

Tabular Equations of State

- By far the most common equation usage of the xRage EOS library is with pressure-temperature tables constructed in a variety of ways
- The pressure-temperature table consists of an array of pressures and an array of temperatures with density and specific internal energy tabulated at the table points

$$\rho_{i,j,m} = \rho(P_i, T_j, m), \qquad e_{i,j,m} = e(P_i, T_j, m)$$

- A key property of these tables for multiple materials is that all materials use the same pressure and temperature grid. (in the above formula m is the index of a material component in the table).
  - All of the material in a table share the same pressure-temperature grid
- The tabular data is extended into a continuous equation of state via the interpolation formulas

$$P_i \le P \le P_{i+1},\ T_j \le T \le T_{j+1}, x = \frac{P - P_i}{P_{i+1} - P_i},\ y = \frac{T - T_j}{T_{j+1} - T_j}$$

$$V(P,T) = \frac{1-y}{(1-x)\rho_{i,j} + x\rho_{i+1,j}} + \frac{y}{(1-x)\rho_{i,j+1} + x\rho_{i+1,j+1}}$$

$$e(P,T) = \left(\frac{e_{i,j}\rho_{i,j}(1-x) + e_{i+1,j}\rho_{i+1,j}x}{\rho_{i,j}(1-x) + \rho_{i+1,j}x}\right)(1-y) + \left(\frac{e_{i,j+1}\rho_{i,j+1}(1-x) + e_{i+1,j+1}\rho_{i+1,j+1}x}{\rho_{i,j+1}(1-x) + \rho_{i+1,j+1}x}\right)y$$

  - It is of some interest to note that these formulas are exact for stiff gamma law equations of state.
  - We also suppressed the material index to save space.

P-T Solver

- The heart of the EOS is to solve the system of equations:

$$e = \sum_{k=1}^{n} \mu_k e_k(P,T)\,, V = \sum_{k=1}^{n} \mu_k V_k(P,T)$$

For a given specific internal energy e, specific volume V, and material mass fractions μ_k, $k = 1, \ldots, n$. Since the mass fractions remain constant during the solve to save space we will suppress these values.

P-T Solver: Step 1, teos_mixed_3

- Find $\tilde{T}$ such that $e = e(P_{\min}, \tilde{T}), T_{\min} \leq \tilde{T} \leq T_{\max}$, extrapolate if needed
  - Let $\tilde{V} = V(P_{\min}, \tilde{T})$
- Use finite differences to approximate the values $\frac{\partial e}{\partial T}, \frac{\partial e}{\partial P}, \frac{\partial V}{\partial T}, \frac{\partial V}{\partial P}$ at $P_{\min}, \tilde{T}$
- Compute
  - The Grüneisen exponent: $\Gamma = V \left.\frac{\partial P}{\partial e}\right|_V = \frac{V\frac{\partial V}{\partial T}}{\frac{\partial V}{\partial P}\frac{\partial e}{\partial T} - \frac{\partial V}{\partial T}\frac{\partial e}{\partial P}}$
  - The isothermal bulk modulus: $\rho c_T^2 = -V \left.\frac{\partial P}{\partial V}\right|_T = -\frac{V}{\frac{\partial V}{\partial P}}$
  - Specific heat at constant volume: $C_V = \left.\frac{\partial e}{\partial T}\right|_V = \frac{\frac{\partial V}{\partial P}\frac{\partial e}{\partial T} - \frac{\partial V}{\partial T}\frac{\partial e}{\partial P}}{\frac{\partial V}{\partial P}}$
- For a stiff gamma law equation of state we have $\rho c_T^2 = P + P_\infty$ so we can fit this EOS model at $P_{\min}, \tilde{T}$ using the above values, if $P_\infty < P_{min}$ set $P_\infty = 0$
- Approximate the solution pressure as $\tilde{P} + P_\infty = C_V \frac{\tilde{T}}{\tilde{V}}$
- If $\tilde{P} \leq P_{\min}$ accept $\tilde{P}$ and $\tilde{T}$ as the EOS solution
- Otherwise proceed to the full solver
- Remark: This step is omitted when the input option use_teos_iter_5 is true

PT Solver: Main Iteration: teos_iter_3

- This is the normal case.
- The input to the P-T equilibrium solver are the specific volume, specific internal energy (V, e), the mass fractions of the material components, and first guess values for the pressure and temperature.
  - Typically, the input pressure and temperature are the values computed by the previous cycle and are often close to the desired solution.
- The solver loops through the following iteration:
  - Given: P^n, T^n, find T^{n+1} such that $e = e(P^n, T^{n+1})$ – uses bisection
  - Perform a Newton update along the iso-energy curve

$$P^{n+1} = P^n + \frac{V - V(P^n, T^{n+1})}{\left(\left.\frac{\partial V}{\partial P}\right|_e\right)(P^n, T^{n+1})}, \left.\frac{\partial V}{\partial P}\right|_e = \frac{\frac{\partial V}{\partial P}\frac{\partial e}{\partial T} - \frac{\partial V}{\partial T}\frac{\partial e}{\partial P}}{\frac{\partial e}{\partial T}}$$

  - Convergence occurs when the values of pressure and temperature don't change within given tolerances, or the energy and specific volume residuals are small enough.
- DETAILS DETAILS DETAILS
  - There are a great many technical complications that must be addressed for a robust solver
    - Handle out of pressure/temperature domain issues – push back to domain or extrapolate
    - Handle numerical singularities in the derivative calculations – test for small denominators
    - Terminate the solution and accept the last computed values if the number of iterations becomes too large

EOS API - xmeos

Evaluating the EOS from a Table

- The code supports the following options for the evaluation of EOS data.

1. Energy-Density evaluations

   Given the mixture component masses (or mass fractions) and the total specific volume and specific internal energy of the mixture compute the mixture pressure and temperature and thermodynamics derivatives (isentropic bulk modulus, density times the Grüneisen export, and specific heat at constant volume).

2. Temperature-Density

   Same as the above except use temperature as input instead of specific internal energy

3. Pressure-Density

   Use pressure as the independent variable

4. Pressure-Temperature

   Use pressure and temperature as the independent variables

- By far option 1 is the most common. The other options are mostly used as part of initialization.

Xmeos - EOS Front End Subroutine

- The main entry point into the equation of state library is the suite of subroutines under the general name of xmeos
- The versions of this routine are implemented via a CPP template and provides support for calls with the list of states to be evaluation in the following ways
  - xmeos_all(EosOpt,options,cell_dim,levs,nummat,mmat,matdef, matident,EosErr,vcell,spvol,cell_sie,prsmx,pres,tev,bmod,dpde,cv, frac_mass,frac_vol,frac_eng,frac_bmod,frac_dpde,frac_cv)
    - The intended use is to evaluate the EOS on all top level cells
  - xmeos_list(EosOpt,options,cell_dim,numtmp,ltmp,nummat,mmat,matdef, matident,EosErr,vcell,spvol,sie,prsmx,pres,tev,bmod,dpde,cv, frac_mass,frac_vol,frac_eng,frac_bmod,frac_dpde,frac_cv)
    - The intended use is to evaluate the EOS at the addresses specified in the argument ltmp
  - xmeos_ary(EosOpt,options,cell_dim,numtmp,ltmp,nummat,mmat,matdef, matident,EosErr,vcell,vol,sie,pmax,pres,tev,bmod,dpde,cv, frac_mass,frac_vol,frac_eng,frac_bmod,frac_dpde,frac_cv)
    - This is an array based version of xmeos_list
- Convenience routines such as xmeos_one are provided to evaluate the EOS at scalars and xmeos_single_mat whichs is specialized for calls with just one material

Xmeos – Arguments explained

- The basic difference between xmeos_all, xmeos_list, and xmeos_ary is the type of arguments used to specify the material component properties.
  - In xmeos_all and xmeos_list these arguments are of type mesh_state_frac_var_t which is a polymorphic type that allows for data compression of the component data
  - In xmeos_ary these are arguments are two dimensional arrays of size cell_dim by nummat
- type(EosOptions_t), intent(in) :: EosOpt
  - This type encapsulates information mostly only meaningful to the EOS internals
- type(EosError_t), intent(out) :: EosErr
  - returns an error status and error message. In normal cases the error value in EosErr should be false
- integer,intent(in) :: cell_dim
  - The value is the allocated length of the array arguments
- integer, intent(in) :: options
  - specifies the independent variables to be used to evaluate the equation of state. It can have one of five values
    - TEMPERATURE_DENSITY_STATE
      - Evaluate the EOS as a function of temperature and specific volume (1/density)
    - PRESSURE_DENSITY_STATE
      - Evaluate the EOS as a function of pressure and specific volume (1/density)
    - PRESSURE_TEMPERATURE_STATE
      - Evaluate the EOS as a function of pressure and temperature
    - NORMAL_ENERGY_DENSITY_STATE, ENERGY_DENSITY_STATE
      - Evaluate the EOS as a function of specific internal energy and specific volume
      - NORMAL_ENERGY_DENSITY_STATE evaluates the material component thermodynamics and is the usual value for this argument

Xmeos – Arguments Explained

- class(levels_t), intent(in) :: levs
  - used to extract material component information
- integer, intent(in) :: nummat
  - the number of material components
- integer, dimension(:), intent(in) :: mmat
  - nummat length array, mmat(m) = material id of material m
- real(REAL64), dimension(:,:), intent(in) :: matdef
  - Material parameters array matdef(n,mmat(m)) is a parameter for material the input material $1 \leq m \leq nummat$
- character(len=*), dimension(:), intent(in) :: matident
  - matident(mmat(m)) is a symbolic name for a material m
- integer,intent(in) :: numtmp
  - The number of EOS evaluations to perform
- integer,intent(in) :: ltmp(:)
  - The indirection array for the EOS's to be evaluated ltmp(n) $1 \leq n \leq numtmp$ is the index in the arrays where data is stored

Xmeos – Arguments Explained

- real(REAL64), dimension(:), intent(in) :: vcell
  - The extensive volumes associated with the EOS states
- frac_mass – component masses or mass fractions
  - xmeos_all, xmeos_list
    - type(mesh_state_frac_var_t) , intent(in) :: frac_mass
  - xmeos_list
    - real(REAL64), dimension(:,:) :: frac_mass
      - Frac_mass(I,n) is the mass of material n) $1 \leq n \leq numtmp$
- real(REAL64), dimension(:), intent(inout) :: pmax
  - The maximum pressure associated with the Lagrangian location for the EOS evaluation
- real(REAL64), dimension(:), intent(inout) :: sie, vol, pres, tev
  - Thermodynamic states. Depending the value of options two will be used as input the other two as output
    - sie - specific internal energy
    - vol – specific volume
    - pres – thermodynamic pressure
    - tev – thermodynamic temperature
- real(REAL64), dimension(:), intent(out) :: bmod, dpde, cv
  - Thermodynamic derivatives
    - bmod – isentropic bulk modulus
    - dpde – derivative of pressure with respect to specific internal energy at constant density
    - cv – specific heat a constant volume

Xmeos – Arguments explained

- Output component values for specific volume and specific internal energy
  - xmeos_all, xmeos_list
    - type(mesh_state_frac_var_t), intent(inout) :: frac_vol, frac_eng
  - xmeos_ary
    - real(REAL64), dimension(:,:), intent(inout) :: frac_vol, frac_eng
- Optional arguments for the component thermodynamic derivatives
  - xmeos_all, xmeos_list
    - type(mesh_state_frac_var_t), intent(inout), optional :: frac_bmod, frac_dpde, frac_cv
  - xmeos_ary
    - real(REAL64), dimension(:,:), intent(inout), optional :: frac_bmod, frac_dpde, frac_cv
- Note: these argument are all really output variables but for technical reasons they are declared intent(inout)